Challenge 8

CY6740 – Machine Learning in CyberSecurity

Tejas Krishna Reddy

NUID: 001423166

30th Nov 2020

Semi-Supervised Learning.

Process:

- Read labeled dataset, remove 'hash' column.
- Divide the dataset into features (X) and labels (y).
- Train the data and check their score using 5- Fold cross validation with each of the 5 models.
- Using trained models, predict labels of unlabeled dataset, then create a merged data to predict scores similar to step 3 for all 5 algorithms.
- Results comparison and reasoning.

Results:

1. Evaluating 5 algorithms cross validation scores on labeled data:

```
clf1 = LinearSVC(class_weight='balanced', random_state = 42, penalty = 'l2')

print("Cross Val Scores of Linear SVC are: ", cross_val_score(clf1, X, y, cv=5))
print("Average Cross val Score for Linear SVC model: ", np.mean(cross_val_score(clf1, X, y, cv=5)))
```

```
Cross Val Scores of Linear SVC are:  [0.54666667 0.56666667 0.54666667 0.57666667 0.62333333]
Average Cross val Score for Linear SVC model:  0.5719999999999998
```

```
clf2 = NuSVC(nu=0.5, kernel = 'rbf', degree = 3, class_weight = 'balanced', random_state = 42)

print("Cross Val Scores of  NuSVC are: ", cross_val_score(clf2, X, y, cv=5))
print("Average Cross val Score for NuSVC model: ", np.mean(cross_val_score(clf2, X, y, cv=5)))
```

```
Cross Val Scores of  NuSVC are:  [0.46333333 0.55333333 0.45666667 0.57       0.52       ]
Average Cross val Score for NuSVC model:  0.5126666666666666
```

```
clf3 = SGDClassifier(penalty = 'l2', learning_rate = 'optimal', random_state = 42)

print("Cross Val Scores of  SGDClassifier are: ", cross_val_score(clf3, X, y, cv=5))
print("Average Cross val Score for SGDClassifier model: ", np.mean(cross_val_score(clf3, X, y, cv=5)))
```

```
Cross Val Scores of  SGDClassifier are:  [0.67       0.66666667 0.66666667 0.67       0.66666667]
Average Cross val Score for SGDClassifier model:  0.6679999999999999
```

```
clf4 = NearestCentroid(metric='euclidean', shrink_threshold=None)

print("Cross Val Scores of  NearestCentroid are: ", cross_val_score(clf4, X, y, cv=5))
print("Average Cross val Score for NearestCentroid model: ", np.mean(cross_val_score(clf4, X, y, cv=5)))
```

```
Cross Val Scores of  NearestCentroid are:  [0.54333333 0.58333333 0.54333333 0.56666667 0.58666667]
Average Cross val Score for NearestCentroid model:  0.5646666666666667
```

```
clf5 = KNeighborsClassifier(n_neighbors=10, weights='uniform', algorithm='auto')

print("Cross Val Scores of  KNeighborsClassifier are: ", cross_val_score(clf5, X, y, cv=5))
print("Average Cross val Score for KNeighborsClassifier model: ", np.mean(cross_val_score(clf5, X, y, cv=5)))
```

```
Cross Val Scores of  KNeighborsClassifier are:  [0.83666667 0.85333333 0.77666667 0.82       0.84333333]
Average Cross val Score for KNeighborsClassifier model:  0.826
```

2. I have written a function which taken in each model and do the steps mentioned in question2:
    - Read unlabeled dataset
    - Predict labels for them
    - Create merged data
    - Evaluate cross val score for the merged data for that particular model

    The function is as follows:

```
def fullDataModel(modl, df):

    ### Read unlabeled dataset
    df1 = pd.read_csv("Dataset_Challenge8/dataset_unlabeled.csv")
    df1 = df1.drop(['hash'], 1)

    # predict the malware
    y_unlabeled = modl.predict(df1)
    df1['malware'] = y_unlabeled

    df.reset_index(drop = True, inplace = True)
    df1.reset_index(drop = True, inplace = True)

    # Merge them together
    mergedData = pd.concat([df, df1], 0)

    # Create X and Y
    target_variable = 'malware'
    X1 = (mergedData.drop([target_variable], 1))
    X1 = MinMaxScaler().fit_transform(X1)
    y1 = (mergedData[target_variable])


    # Print their scores
    try:
        int(cross_val_score(modl, X1, y1, cv=5)[0])
        print("Cross Val Scores of are: ", cross_val_score(modl, X1, y1, cv=5))
        print("Average Cross val Score for model: ", np.mean(cross_val_score(modl, X1, y1, cv=5)))
    except:
        print("Average Cross val score for model: ", modl.score(X1, y1))
    pass
```

Now the results are as follows for each algorithm:

Linear SVC Model

```
fullDataModel(modl1, df)
```

```
Cross Val Scores of are:  [0.68784658 0.76259242 0.7536684  0.74962449 0.75563258]
Average Cross val Score for model:  0.7418728956903097
```

NuSVC Model

```
fullDataModel(modl2, df)
```

```
Average Cross val score for model:  0.9122859717632923
```

SGD Classifier:

```
fullDataModel(modl3, df)
```

```
Cross Val Scores of are:  [0.80360444 0.80360444 0.8034662  0.80358174 0.80358174]
Average Cross val Score for model:  0.8035677132554024
```

Nearest Centroid Classifier

```
fullDataModel(modl4, df)
```

```
Cross Val Scores of are:  [0.86286969 0.86668207 0.87521664 0.86909301 0.87533218]
Average Cross val Score for model:  0.8698387163083534
```

KNN Classifier

```
fullDataModel(modl5, df)
```

```
Cross Val Scores of are:  [0.98555915 0.98625231 0.9859041  0.9859041  0.9864818 ]
Average Cross val Score for model:  0.9860202932071577
```

3. In step 1, we could see that majority of the scores ranged from 0.5 to 0.7. Whereas, in Step 2 we could observe that mostly in between 0.8 to 0.98. Hence, we could see a significant improvement in the overall score by using only labeled data and by using merged Data.

   Linear SVC showed a ~ +0.25 score increase. NuSVC has a very significant improvement over ~ +0.4.
   SGD classifier and Nearest centroid Classifiers had a boost of approximately +0.3 increase. KNN algorithm was doing fairly better with the labeled data itself, however it improved by +0.1 and came close to a perfect classification with the merged Data.

Challenge 8 - Semi Supervised Algorithms

- Tejas Krishna Reddy
- 30th November 2020

In [1]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

Read the labeled dataset

In [3]:
```
df = pd.read_csv('Dataset_Challenge8/dataset_labeled.csv')

df = df.drop(['hash'], 1)
df.head(4)
```

Out[3]:

	size_of_data	virtual_address	entropy	virtual_size	malware
0	443392	4096	6.529624	442984	1
1	331264	4096	6.604314	330784	1
2	74240	4096	6.046789	73819	1
3	219648	4096	6.497018	219524	1

Divide the Dataset into test and train.

In [4]:
```
target_variable = 'malware'

X = (df.drop([target_variable], 1))
y = (df[target_variable])
```

Scaling

In [5]:
```
### Scaling
from sklearn.preprocessing import MinMaxScaler, StandardScaler
X = MinMaxScaler().fit_transform(X)
```

Modelling the labeled data

Modeling the labeled data

In [6]:
```
from sklearn.svm import LinearSVC, NuSVC
from sklearn.linear_model import SGDClassifier
from sklearn.neighbors import NearestCentroid, KNeighborsClassifier
from sklearn.model_selection import cross_val_score
```

In [7]:
```
clf1 = LinearSVC(class_weight='balanced', random_state = 42, penalty = 'l2'

print("Cross Val Scores of Linear SVC are: ", cross_val_score(clf1, X, y, c
print("Average Cross val Score for Linear SVC model: ", np.mean(cross_val_s
```

```
Cross Val Scores of Linear SVC are:  [0.54666667 0.56666667 0.54666667 0.57
666667 0.62333333]
Average Cross val Score for Linear SVC model:  0.5719999999999998
```

In [8]:
```
clf2 = NuSVC(nu=0.5, kernel = 'rbf', degree = 3, class_weight = 'balanced',

print("Cross Val Scores of  NuSVC are: ", cross_val_score(clf2, X, y, cv=5)
print("Average Cross val Score for NuSVC model: ", np.mean(cross_val_score(
```

```
Cross Val Scores of  NuSVC are:  [0.46333333 0.55333333 0.45666667 0.57
0.52      ]
Average Cross val Score for NuSVC model:  0.5126666666666666
```

In [9]:
```
clf3 = SGDClassifier(penalty = 'l2', learning_rate = 'optimal', random_stat

print("Cross Val Scores of  SGDClassifier are: ", cross_val_score(clf3, X, 
print("Average Cross val Score for SGDClassifier model: ", np.mean(cross_va
```

```
Cross Val Scores of  SGDClassifier are:  [0.67       0.66666667 0.66666667
0.67       0.66666667]
Average Cross val Score for SGDClassifier model:  0.6679999999999999
```

In [10]:
```
clf4 = NearestCentroid(metric='euclidean', shrink_threshold=None)

print("Cross Val Scores of  NearestCentroid are: ", cross_val_score(clf4, X
print("Average Cross val Score for NearestCentroid model: ", np.mean(cross_
```

```
Cross Val Scores of  NearestCentroid are:  [0.54333333 0.58333333 0.5433333
3 0.56666667 0.58666667]
Average Cross val Score for NearestCentroid model:  0.5646666666666667
```

In [11]:
```
clf5 = KNeighborsClassifier(n_neighbors=10, weights='uniform', algorithm='a

print("Cross Val Scores of  KNeighborsClassifier are: ", cross_val_score(cl
print("Average Cross val Score for KNeighborsClassifier model: ", np.mean(c
```

```
Cross Val Scores of  KNeighborsClassifier are:  [0.83666667 0.85333333 0.77
666667 0.82       0.84333333]
Average Cross val Score for KNeighborsClassifier model:  0.826
```

In [ ]:
```
```

Train the Models using Labeled Data

In [12]:
```
# train models using labeled data
modl1 = LinearSVC(class_weight='balanced', random_state = 42, penalty = 'l2
modl2 = NuSVC(nu=0.5, kernel = 'poly', degree = 3, class_weight = 'balanced
modl3 = SGDClassifier(penalty = 'l2', learning_rate = 'optimal', random_sta
modl4 = NearestCentroid(metric='euclidean', shrink_threshold=None).fit(X, y
modl5 = KNeighborsClassifier(n_neighbors=10, weights='uniform', algorithm='
```

Define a function to:

- Read and preprocess unlabeled data.
- Predict y for unlabeled data.
- Merge labeled and unlabeled data to get Merged Data.
- Predict cross val score against Merged Data for that model.

In [15]:
```
def fullDataModel(modl, df):

    ### Read unlabeled dataset
    df1 = pd.read_csv("Dataset_Challenge8/dataset_unlabeled.csv")
    df1 = df1.drop(['hash'], 1)

    # predict the malware
    y_unlabeled = modl.predict(df1)
    df1['malware'] = y_unlabeled

    df.reset_index(drop = True, inplace = True)
    df1.reset_index(drop = True, inplace = True)

    # Merge them together
    mergedData = pd.concat([df, df1], 0)

    # Create X and Y
    target_variable = 'malware'
    X1 = (mergedData.drop([target_variable], 1))
    X1 = MinMaxScaler().fit_transform(X1)
    y1 = (mergedData[target_variable])


    # Print their scores
    try:
        int(cross_val_score(modl, X1, y1, cv=5)[0])
        print("Cross Val Scores of are: ", cross_val_score(modl, X1, y1, cv
        print("Average Cross val Score for model: ", np.mean(cross_val_scor
    except:
        print("Average Cross val score for model: ", modl.score(X1, y1))
    pass
```

In [ ]:
```
```

Linear SVC Model

In [16]:
```
fullDataModel(modl1, df)
```

```
Cross Val Scores of are:  [0.68784658 0.76259242 0.7536684  0.74962449 0.75
563258]
Average Cross val Score for model:  0.7418728956903097
```

NuSVC Model

In [17]:
```
fullDataModel(modl2, df)
```

```
Average Cross val score for model:  0.9122859717632923
```

SGD Classifier:

In [18]:
```
fullDataModel(modl3, df)
```

```
Cross Val Scores of are:  [0.80360444 0.80360444 0.8034662  0.80358174 0.80
358174]
Average Cross val Score for model:  0.8035677132554024
```

Nearest Centroid Classifier

In [19]:
```
fullDataModel(modl4, df)
```

```
Cross Val Scores of are:  [0.86286969 0.86668207 0.87521664 0.86909301 0.87
533218]
Average Cross val Score for model:  0.8698387163083534
```

KNN Classifier

In [20]:
```
fullDataModel(modl5, df)
```

```
Cross Val Scores of are:  [0.98555915 0.98625231 0.9859041  0.9859041  0.98
64818 ]
Average Cross val Score for model:  0.9860202932071577
```

In [ ]:
```

```